A prototyping platform based on a PCI micronetwork and Leon multiprocessor system

Florent Berthelot, Fabienne Nouvel, Dominique Houzet

A Prototyping Platform based on a PCI Micronetwork and Leon multiprocessor system

Florent Berthelot , Dominique Houzet , Fabienne Nouvel
CNRS UMR 6164 IETR/INSA Rennes 20 av des Buttes de Coesmes, 35043 Rennes, France
florent.berthelot@ens.insa-rennes.fr {dominique.houzet , fabienne.nouvel}@insa-rennes.fr

Abstract

System-on-chip (SoC) designs provide integrated solutions to challenging design problems in the telecommunication, multimedia, and consumer electronics domains. Much of the progress in these fields hinges on the designers ability to conceive complex electronic engines under strong time-to market pressure. Success will rely on using appropriate design and process technologies, on the ability to interconnect existing components including processors, controllers, and memory array reliably as well as on the capability to validate such complex designs.
This last point is the topic of this paper which presents a prototyping methodology and flexible hardware platform developed to facilitate rapid prototyping and validation of such telecommunication systems.

1. Introduction

The transition to SoC design is not straightforward and brings different challenges. While systems complexity is growing, the time-to-market window is decreasing and the simulation times become prohibitive. This can be dealt in several ways: by raising the design level of abstraction, by reusing IP and by the use of real-time emulation platforms. The first solution requires good high-level synthesis tools and description languages. The second is accomplished by socketization. Thus, standards are very useful.

VSIA [1] releases on-chip bus (OCB) interface standard to facilitate design reuse hence accelerating design architectures exploration and synthesis methodologies on a real-time prototyping platform. Software drivers and glue logic are generated to connect processors to peripheral devices, hardware co-processors, or communication interfaces.

We are using this prototyping platform and methodology in the framework of the 4-More IST European project [2], in which our laboratory is widely involved. The project purpose is a twofold study: transmission techniques for integrated broadband cellular systems, eg. MC-CDMA, and SoC architecture design for high performance demanding telecom systems.

The first part of this paper describes the hardware prototyping platform based on a configurable Network-on-Chip (NoC) [3] implemented with the PCI bus. In fact here the NoC is a hierarchical bus. The prototyping platform is used to validate the whole SoC designed around the proposed NoC, with a real-time execution. Numerous papers have described solutions to help the design of SoC based on micronetwork [4].

The second part presents LEON processor [5] which acts as software component on our prototyping platform. Our work is focused on its architecture simplification and adaptation to build a multiprocessor system which communicates through PCI OCB. An implementation example using 3 LEON processors with VCI interfaces and PCI backbone is presented and compared with an AMBA-AHB [6] OCB which is widely diffused in SoC designs.

2. Related Work

Hardware-software co-design approaches differ from one another by their target architectures and by their objectives.

The synthesis from specification allows integration of application-specific and programmable components. High-level descriptions of system are refined into a physical implementation. The focus is put on one of the following points: interfacing, partitioning, refinement, simulation and synthesis. The platform approach has a goal to map an application into a pre-designed complex SoC architecture, by configuring and extending this architecture.

The last solution, IP assembly, allows reuse without redesign work. This ability lets designers consider IP cores as autonomous plug-and-play subsystems in subsequent designs. Designing SoCs based upon reusable IP cores is essential for meeting stringent time-to-market requirements. Once suitable IP cores are identified, the focus shifts to the integration challenge: how to build a working system from a collection of generic and domain-specific cores that were not designed to work together [7, 8]. One of the key issues in hardware/software co-design is thus the generation of suitable interprocess communication interfaces, and to determine the amount and type of communication between different components of a digital system. The problem of communication is one of the main obstacles to obtain full benefit of high performance components. In all cases, validation of the physical implementation is a critical point which can be addressed with emulation. A complete hardware/software system can be emulated as an architecture–precise prototype on a real-time prototyping

platform [9] to verify the derived communication architecture as well as the hardware/software partitioning.

3. Design methodology

Our prototyping platform is aimed to be used with a System–level language like SystemC [10]. During the co-design phase the system model is partitioned with VCC tools into embedded software for the host processor and the LEON embedded processor and hardware code for the FPGAs. We use Gcc compiler for the development of embedded software for the target host and LEON processors. Standard software development tools, like GDB with DDD, are also used for debugging this software system partitioning and get optimized code for the host and LEON processors.

In the system level design flow for hardware system partitioning, the SystemC compiler and FPGA compiler tools translate an input behavioural SystemC description into a target technology gate-level netlist. The netlist is then placed and routed by standard Xilinx layout tools, which produce a full FPGA configuration bitstream. In the same time, we generate a hardware interface for each implemented component. The final product of the design flow is a set of binary files representing programs for host processor, LEON processors and the FPGAs. These can be loaded onto the respective components of the prototyping platform to build a prototype with a real-time communication system.

We have chosen LEON processor for many reasons. It is an open, portable and non-proprietary processor design with high performance. Its scalability and modular design particularly match prototyping requirements.

Communication between heterogeneous components can be made through classic busses such as PCI, USB, or On Chip Busses such as AMBA, CoreConnect, etc. This imposes the existence of a wrapper on each side of the bus. Nearly every embedded processor has its own proprietary system bus. Some have defined proprietary peripheral busses as well. This situation can makes it difficult to take a Virtual Component designed for a SOC with one embedded processor and move it to a different chip. A few busses (e.g., AMBA buses for ARM processors) are widely supported and can be considered a de facto standard in some application fields.

One interesting option for OCB is an on-chip version of the PCI-X bus. In this first study we have evaluated this interest of the PCI 2.2 bus. The choice of the PCI bus is guided by the maximal re-use of wrappers, boards, test equipments available for the widespread PCI bus. With that bus, it is possible to build a low-cost hardware prototyping platform which integrates the busses which will be implemented in the final SoC. The main idea is to provide a prototyping platform which integrates physically the SoC NoC.

Our physical platform uses PCI for the SoC OCB as well as for the external bus. The PCI wrapper behaves like a bridge by translating the signals between the PCI bus and the final chip. (See figure 1.). The wrapper is quite specific to each bus and it must have a compatible interface with any type of component, possibly a VCI interface. This approach requires to provide two wrappers for each side of the communication bus: a device driver (soft) for software implementation using a microprocessor and a bus controller IP (hard) which is compatible with VCI interface.

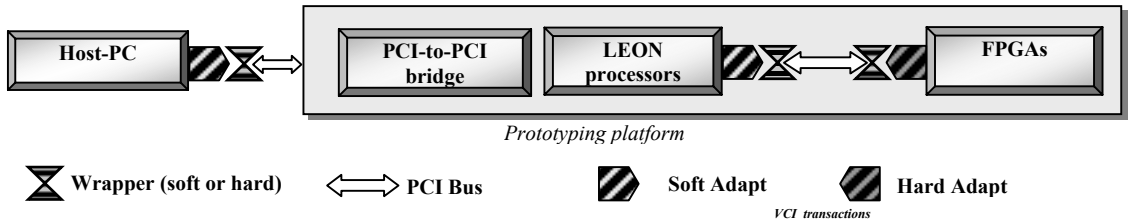
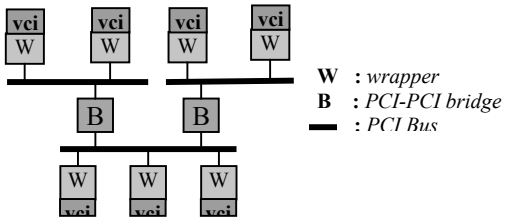


Fig. 1. Prototyping platform components communications fashion

Fig. 2. micronetwork architecture



Our platform is based on a micronetwork infrastructure which implements the lower level layers of the communication stack protocols. Each layer of the whole micronetwork is configurable in terms of architecture, that is bus numbers, sizes, frequencies and topology organization.

An overall overview of the micro-network architecture used in our prototyping platform is presented in Figure 2.

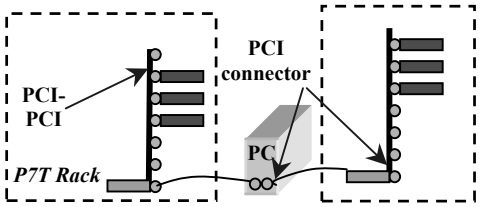


Fig. 3. 2-rack platform

This platform can be configured to match the structure of the SoC in terms of busses sizes and organization. In that way, the entire SoC is managed and tested in real-time from a PC host computer through its PCI bus.

This platform consists of a hierarchy of PCI racks receiving seven 8-FPGA boards each. Figure 3 presents an example of a 2-level hierarchical platform with two racks

connected to a host computer which is used for test purposes or for the modelling of a computer application software part. We are using 8 Xilinx XC2V6000 FPGAs connected via the PCI bus. An example of a SoC floorplan designed around the PCI NoC with the platform presented here is shown in figure 4.

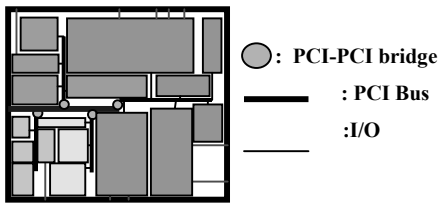


Fig. 4. SoC floorplan

The external PCI bus is visible on the left side of the chip. This example is build around a 2-level hierarchical PCI-bus with four busses at the second level of hierarchy. The platform used to model this SoC is configured with one rack of four 8-FPGA boards. Each IP in the SoC is emulated in one of the FPGA. A first rough estimation of size and performances between FPGA and ASIC has been elaborated.

4. LEON's integration on prototyping platform

The LEON processor is a synthesisable VHDL model of a 32-bit processor compliant with the SPARC V8 architecture. The model is highly configurable (register windows, instruction and data caches), and particularly suitable for system-on-a-chip (SOC) designs. The full source code is available under the GNU LGPL license. Leon reference model architecture includes AMBA-2.0 AHB and APB on-chip busses, hardware multiply, divide and MAC units, separate instruction and data caches (Harvard architecture) , a full master/target interface based on the PCI bridge from OpenCores, with an additional AHB interface. (See figure 5.)

For the integration in our platform we have chosen to simplify the previous architecture and discard AMBA bus utilization, which isn't our prototyping platform communication medium.

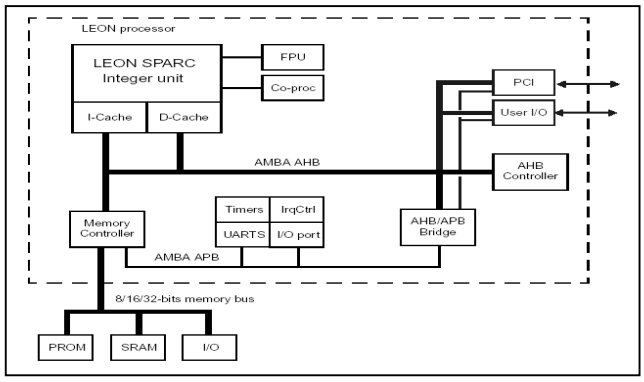


Fig. 5. Leon reference model architecture

Our proposed SoC architecture is based on a micro-network with LEON processors modified with distinct program and data memory access, as presented in figure 6. A dedicated RAM access for DMA transfers between LEONs through PCI bus is provided. Hence instruction and data accesses are local on each processors. PCI bus is reserved for inter-Leons communications or with others modules. The software adaptation layer is represented by the PCI interface module, The wrapper which has access to the bus is integrated in the PCI-IP. Both modules have VCI interfaces. After the compilation phase of C/C++ applications for LEONs, we obtain memory initialisation files (mif) for each ROM. Theses files are then used during synthesis phase of the whole design.

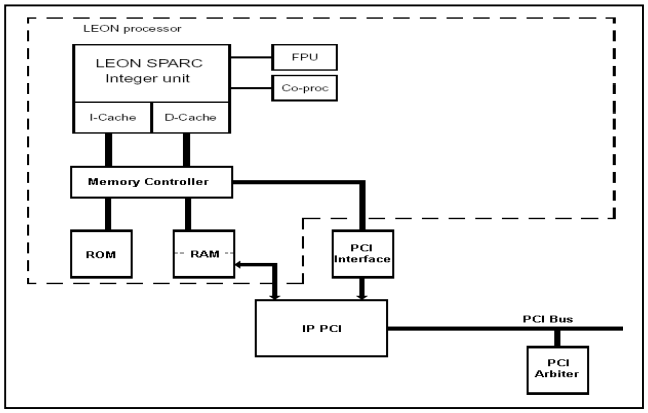


Fig. 6. Modified Leon architecture for PCI

We have designed a second version which operates with AMBA-AHB bus. So we can compare performance of these two models and justify our choice for a PCI-OCB based prototyping platform. This version is illustrated in figure 7.

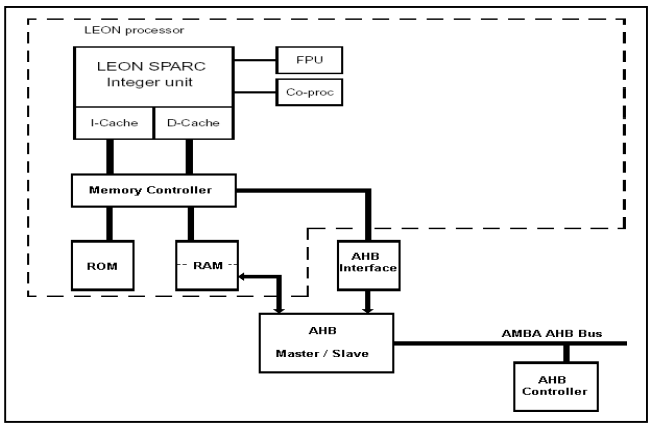


Fig. 7. Modified Leon architecture for AMBA AHB

5. Implementation Results

Two micro-network models based upon PCI and AMBA-AHB backbone have been synthesized and compared. The results in terms of timing and logic resources are summarized in the following tables. These features are obtained with Xilinx Virtex FPGA(xc2v6000-6) and Leonardo Spectrum synthesis tool. Tables 1 and 2 summarize these results. All the processor memories (instruction / data caches and ROM/RAM memories) are implemented with BlockRam Virtex Technology.

Our AMBA-AHB model is designed for a straightforward DMA transfer between LEONs without waitstates or interrupt considerations. This two kind of on-chip busses differ in their bus arbiter/decoder way of implementation.

	LUT4	FF
IP PCI agent	1007	314
PCI arbiter	15	7

Table 1: *- Logic resources -PCI model (32 bits)*

	LUT4	FF
AMBA AHB agent	478	332
AMBA AHBarbiter / decoder	278	11

Table 2: *- Logic resources -AMBA AHB model (32 bits)*

FF is for D Flip-Flop LUT4 is for 4 entries logic gates

The bus decoder, which selects a slave targeted by a master request, is embedded in each PCI agent, whereas this function is located within bus arbiter module for AMBA-AHB bus.

Timing results for one-word transaction is clearly in advantage of AMBA-AHB version. A more optimized version of PCI-IP module, particularly for the bus decoder functionality can achieve same results. Timing features of AMBA-AHB or PCI based models are similar for wide data block transfers by the use of large burst requests. Bandwidth results are calculated for a 100 MHz bus frequency, which is obtained from our synthesis results for a micro-network based on three LEONs. The following tables summarize these results.

One-word (32-bit)			
Latency		Bandwidth (100 MHz)	
write	read	write	read
12 clk	10 clk	31 MB/s	37 MB/s

Table 3: - Single word - PCI model features

1Kword message (32-bit)			
Latency		Bandwidth (100 MHz)	
write	read	write	read
1K+11 clk	1K+9 clk	396 MB/s	397 MB/s

Table 4: *- 1K word message -PCI model features*

One-word (32-bit)		1Kword message (32-bit)	
latency	Bandwidth (100 MHz)	latency	Bandwidth (100 MHz)
Read/write	Read/write	Read/write	Read/write
5 clk	66 MB/s	1K+4 clk	399 MB/s

Table 5: *- AMBA AHB model features -*

Virtex II 6000	% Slices	Leons frequency	Bus frequency
3 LEON / PCI bus	23	46 MHz	86 MHz
3 LEON /AMBA-AHB bus	22	46 MHz	76 MHz

Table 6: *- Micro-network main features -*

Table 6 summarizes integration results of a micronetwork composed of 3 LEONs. PCI version has a better bus frequency which is due to non-pipelined bus operations, hence involving a lower design complexity, unlike pipelined AMBA-AHB 2.0 that performs operations concurrently in order to increase the throughput.

Results on LEONs implementation show the possibility to integrate up to 4 LEONs on each FPGA with PCI backbone. Each processor can be configured, in term of hardware architecture, to match software applications.
These results demonstrate the ability of PCI to operate as an on-chip bus in the framework of our project, hence providing a direct and simple way of communication with the host-PC.

6. Conclusion

In this paper we have described a prototyping platform used to design large and complex systems on a single chip. This platform concepts answer the need to amortize the enormous engineering cost involved in designing and testing large chips and by the demand for easy-to-use methods to exploit the parallel processing capacity provided by multiple computational IP resources.

We have described a physical prototyping platform which implements the PCI based NoC and the whole SoC under design, with expandable low-cost racks and FPGA boards. First results have shown its ability to efficiently deal with high performances telecom applications. The results presented here confirm the ability to use PCI as an OCB, with the main advantage of high availability of design and test tools and boards for this widespread bus.

Applications using RTEMS [11] executive services can be compiled for LEON processors. We are working on the development of a multiprocessor communications interface layer (MPCI), which enable processors to communicate with one another on the OCB through RTEMS multiprocessing capabilities. We obtain a real-time multiprocessor system. Furthermore a large part of software and hardware components of this prototyping platform are free source codes and based on a non-proprietary OCB. We also plan to integrate the PCI-X standard for more efficient communications and backward compatibility.

7. References

[1] Virtual Component Interface Standard (OCB 2 1.0). On-Chip Bus Development Working Group, March 14, 2000

[2] S. KAISER, Y. DURAND, L. HERAULT, J-F. HELARD, D. MOTTIER, A. GAMEIRO, J.ARQUINERO, F. BERENS.
"4G MC-CDMA multi antenna system on chip for radio enhancements (4MORE)",
IST Summit Conference, Lyon, France, 28-30 June 2004.

[3] L. Benini, G. De Micheli, “Networks on Chip: A New SoC Paradigm”, IEEE Computer, 2002.

[4] M. Vasilko et al., “A Rapid Prototyping Methodology and Platform for Seamless Communication Systems”, IEEE RSP01 Workshop, Monterey, June 2001.

[5] Leon / AMBA VHDL model description,
European Space Agency (www.estec.esa.nl/wsmwww/leon)

[6] AMBA Specification (Rev 2.0), ARM (www.arm.com)

[7] S. Nrarayan and D. Gajski. Interfacing in-compatible protocols using interface process generation. Proc of DAC 1995.

[8] G.Cyr, G.Bois and M.aboulhamid. synthesis of communication interfaces for SoC using VSIA recommendations, Proc. of DATE 2001 conference.

[9] Embedded system Architecture design based on Real-Time Emulation. C. Nitsch, K. Weiss, T. Steckstor, W. Rosenstiel,11th IEEE International Workshop on rapid prototyping (RSP), USA, 2000

[10] K. Bartelson A New Standard for System Level Design, September 1999, (www.systemC.com).

[11] RTEMS: Real Time Executive for Multiprocessor Systems (www.rtems.com)